

NASA-CR-200018

NASA/WVU Software IV & V Facility  
Software Research Laboratory  
Technical Report Series

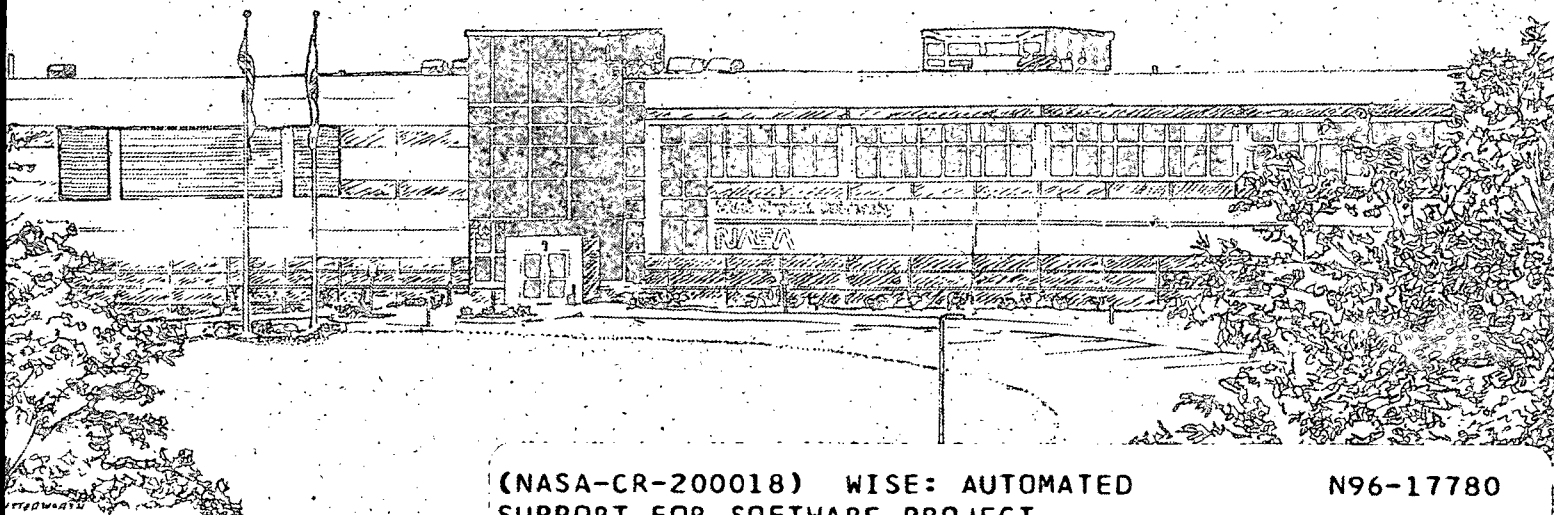
NASA-IVV-95-001  
WVU-SRL-95-001  
WVU-SCS-TR-95-21  
CERC-TR-TM-95-008

*NECW-0040*

**WISE: Automated Support for Software Project  
Management and Measurement**

by Sudhakar Ramakrishnan

*IN-61-CR  
7262  
p-63*



(NASA-CR-200018) WISE: AUTOMATED  
SUPPORT FOR SOFTWARE PROJECT  
MANAGEMENT AND MEASUREMENT M.S.  
Thesis (West Virginia Univ.) 63 p

N96-17780

Unclass

G3/61 0098250



National Aeronautics and Space Administration



West Virginia University

---

# **WISE: Automated Support for Software Project Management And Measurement**

**Sudhakar Ramakrishnan**

*B.S., Anna University, India, 1993  
August 1995*

Department Of Computer Science  
West Virginia University  
Morgantown, WV-26505

*Submitted in partial fulfillment of the requirements  
for the degree of Masters of Science*

**Thesis Committee:**  
John R. Callahan, Chair  
Raghu Karinithi  
Murali Sitaraman

This research was supported part by NASA-IVV Cooperative Research Project agreement, in part by Concurrent Engineering Center (CERC), NASA-IVV Facility Fairmont.

The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the NASA-IVV facility, CERC.

---

APPROVAL OF EXAMINING COMMITTEE

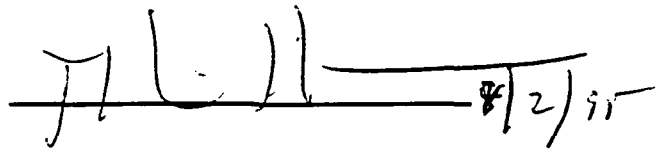
This thesis for the Masters of Science degree by  
*Sudhakar Ramakrishnan*  
has been approved for the  
Department of Computer Science  
by



*John R. Callahan, Ph.D., Chair*



*Raghu Karinithi, Ph.D.*



*Murali Sitaraman, Ph.D.*

4/2/95  
Date

To my grandpa, my parents and my brother

## Acknowledgments

---

I would like to thank Professor John R. Callahan for the guidance, criticism, and experience needed to generate worthy research; for areas of implementation meant only for the most serious hackers in the universe; and for providing an environment for research which was both stimulating and fun and helped me to look into the making of WISE in a profoundly new way.

I am obliged to my thesis committee - John R. Callahan, Raghu Karinthi, and Murali Sitaraman - for their helpful commentary, advice, guidance and patience. I thank professor Murali for asking questions about how I think, for leading me to new questions, and for the valuable discussions we had as a part of the Advanced Reuse Course. I extend my appreciation to Professor Gerald Hobbs for his course on Statistics, which gave me insight into statistical analysis and process control.

I thank Jason Ikkai at NCSA and James Pitkow at Georgia Tech for their help with GSQL and GSQL-ORACLE. I wish to thank all of my fellow students, past and present members of the NASA-IVV research team and my friends for their support and friendship during my stay at WVU. Thank you Anca, Calin, Chen, George, Kiran, Konki, Mittal, Muni, Nanda, Raja, and Sethu. Thanks to the ACM trio - Paul, Laurent, and Saveen - for their invaluable friendship. Special thanks go to Anand and Rahul, who helped mold my research through discussions, for their inspiration and support. Thanks to our research group - Jeff, Todd, Wei, and Wu. A very talented group, they gave useful feedback. I especially thank Todd, for setting an example of how to work out a vision towards practical results. I also thank a number of people at the NASA IV&V facility in Fairmont and CERC.

Finally, my deepest appreciation goes to some very special people in my life. I am grateful to my grandpa, my parents, and my brother for their incredible support over the years. Without them there would be no research, no thesis. I owe them everything. And all my cousins and relatives have given me their inspiration, support, and the reassurance that it was worth all the effort.

---

## Contents

<b>Acknowledgements .....</b>	<b>iv</b>
<b>List Of Figures .....</b>	<b>vii</b>
<b>Abstract .....</b>	<b>viii</b>
<b>1 Introduction.....</b>	<b>1</b>
1.1 Managing Projects .....	1
1.2 The Thesis .....	2
1.2.1 Motivation .....	3
1.2.2 Project Management on the Web.....	5
1.2.3 Organization of this document.....	5
<b>2 System in a Nutshell .....</b>	<b>7</b>
2.1 The need for Software Process Improvement .....	9
2.2 Role of process improvement tools .....	10
2.3 Shaping the WISE project Management Tool .....	11
2.4 Prototypes .....	12
2.5 Integration of Metrics .....	13
2.6 Overview of System Working .....	14
2.7 Project Environment .....	16
<b>3 WISE Architecture .....</b>	<b>17</b>
3.1 Overview of Tool Working.....	18
3.2 Supporting To-do Items .....	19
3.3 Database Connectivity .....	19
3.4 Statistical Analysis Support .....	21
3.5 Graphical Support .....	21
<b>4 Wise in action .....</b>	<b>23</b>
4.1 Life cycle of WISE .....	23
4.2 Startup.....	25
4.3 WISE Main Page .....	27
4.4 Viewing an Issue.....	28
4.5 New Issue .....	28
<b>5 Related Work .....</b>	<b>32</b>

---

---

5.1 Software Automation and Measurement-Strong As Ever .....	32
5.2 Systems .....	36
<b>6 Conclusions.....</b>	<b>41</b>
<b>Bibliography .....</b>	<b>48</b>
<b>Appendix A.....</b>	<b>51</b>
Software Internals .....	51

## List of Figures

Figure 1. WISE Architecture.....	18
Figure 2. WISE Startup.....	24
Figure 3. WISE Main Page.....	26
Figure 4. WISE Issue View .....	27
Figure 5. WISE Issue Insertion .....	29
Figure 6. WISE Metrics .....	31



## **WISE: Automated Support for Software Project Management And Measurement**

by

*Sudhakar Ramakrishnan*

Submitted to the Department of Statistics and Computer Science on July 20, 1995 in partial fulfillment of the requirements for the degree of Masters of Science

### **Abstract**

One important aspect of software development and IV&V is measurement. Unless a software development effort is measured in some way, it is difficult to judge the effectiveness of current efforts and predict future performances. Collection of metrics and adherence to a process are difficult tasks in a software project. Change activity is a powerful indicator of project status. Automated systems that can handle change requests, issues, and other process documents provide an excellent platform for tracking the status of the project. A World Wide Web based architecture is developed for (a) making metrics collection an implicit part of the software process (b) providing metric analysis dynamically (c) supporting automated tools that can complement current practices of in-process improvement (d) overcoming geographical barrier. An operational system (WISE) instantiates this architecture allowing for the improvement of software process in a realistic environment. The tool tracks issues in software development process, provides informal communication between the users with different roles, supports to-do lists (TDL), and helps in software process improvement. WISE minimizes the time devoted to metrics collection, analysis, and captures software change data. Automated tools like WISE focus on understanding and managing the software process. The goal is improvement through measurement.

---

---

# Chapter 1

## Introduction

---

This thesis is concerned with the development of the first World-Wide-Web-based<sup>1</sup> based tool to help in the automation of software project management and measurement. Such an automated tool can transform chaotic software development projects into more controlled and manageable processes. The thesis also examines how metrics collection can be made an implicit part in project management and execution. The software process must improve continuously to produce software products and improve the capability of the organization to produce better products.

### 1.1 Managing projects

---

Automated support plays an important role in the software process improvement. Management refers to a whole array of tasks including status reporting, change management. Handling of change requests, problem reports, activity log entries, maintaining record of changes, bugs in software development effort become

---

<sup>1</sup>World Wide Web is officially described as a "wide-area hypermedia information retrieval initiative aiming to give universal access to a large universe of documents".

---

quite complicated even in a small group. Multiple releases of software bring issues that make issue handling even more critical.

One of the important aspect of software development and IV&V (Independent Verification and Validation) is *measurement*. Collection of metrics and adherence to a process are difficult tasks in a software project. Watts Humphrey in his book Managing the Software Process [WATTS90] states:

The most disastrous mistakes are often made when the project is under the greatest schedule pressure. These are often caused by a loss of change control that started with a quick object patch. While the harried programmers invariably intended to document their changes whenever there was time, it is extremely difficult to remember precisely what was done and why.

Unless a software development effort can be measured in some way, it is difficult to judge the effectiveness of current efforts and predict future performance. Software engineers need tools and techniques to minimize their metric duties and reduce development costs that accrue due to metric collection. Tools that support analysis at the project level are required. Separate metrics group should not collect metrics and analyze them for the developers. Rather, the developers should be able to access the measurements, evaluate them and take action [SHAR93].

## 1.2 The Thesis

---

Automated systems that can handle change requests, issues, problem reports, activity log entries and other process documents provide an excellent platform for tracking the status of the project. The WISE (Web Integrated Software Project

---

Management and Metrics Tool) helps developers keep track of problems, maintain a record of changes, solutions, issues, and errors. Such an automated effort helps developers coordinate the work of a project, distribute effort, track the discrepancies, log of all entries, and maintain a flow of data between the various groups.

This thesis takes a step towards providing an automated support environment for software project management and measurement and highlights the benefits of such an effort. Metrics collection is made an implicit process in the project management. Using WISE an entire software development team can examine the performance of their own software process, how they can make their process better, and where they need to begin in product activities. Communications between the various members of the development team are structured and made more explicit.

### 1.2.1 Motivation

---

WISE started as an idea to put a programmers "to-do" list (TDL) on the WWW and allow programmers and V&V groups to view the project metrics. The aim of V&V is to help monitor the development or maintenance of organization's performance. There is a need to ensure that established standards and procedures are followed to produce products according to plan while simultaneously improving the organization's capability to produce better products. There is a need to mature chaotic software development projects into more controlled and manageable processes. Current software process have to be defined and evaluation standards are applied in a meaningful fashion. It is important to measure the software process because currently there does not exist a means of determining whether we are improving our software processes or not. To

---

establish goals for improvement, the current status of software development must be understood. Measurement is the key solution to establish a process baseline from which improvements can be assessed. Based on such measurements the V&V team can give better insight into the current state of the project.

Statistical control in measurement of software process is of growing importance because "one can only control what one can measure". Overzealous measuring, however can prove to be disruptive. Software metrics contain enormous information about a project: including components, change requests, the testing information, time spent on the tests, computer hours, project estimates, different phase dates for completion of tasks, maintenance reports, development status of each group. All these metrics when analyzed and put in a proper way can help in the V&V team estimate the projects current status.

The WISE tool helps initiate effective software process changes. WISE automates metric collection implicitly in a software process and displays a project's current status in a continuous fashion. This approach has proved helpful.

While making design plans for such a tool, we encountered many discussions on the html work group for the need of a issue-tracking tool for their HTML 2.0 specifications. A need for a tool that would embody a mechanism to maintain a tree of issues was needed to maintain issues for large projects. The idea of having the status of all issues in a nutshell gave rise to metrics. These metrics would provide guidance for better decision making. The *availability* of WWW browsers and *accessibility across all platforms* makes such a tool highly versatile.

While surveying existing tools and studying their functionality, one thing we noticed is that none of them provided a method by way of which the entire software

---

process change is recorded. WISE, we thought would serve to improve the software process by providing a repository of changes made to each issue over time.

### 1.2.2 Project Management on the Web

---

The Web has become the application platform of choice for many enterprises. The World Wide Web is based on a client server architecture. The WWW consists of documents, links, indexes which may be searched. Gateways are servers that make existing information systems visible on the web. The WWW brings the advantage of client server architecture at a very cost and supports the concept of rapid application deployment.

A WWW-based system for allowing access to project documents and issue reports that helps control and manage the development process better. Users can link project artifacts such as documents and code generated during the life cycle of a software system. Example of such document could be the requirement diagrams, design diagrams. CERN is working on a production type for LIGHTS document layer for a program used by physicists to reconstruct collisions in a detector.

### 1.2.3 Organization of this document

---

This thesis has been distributed into 6 chapters. Chapter 2 titled System in a nutshell gives an view of the whole automated effort and talks about software process improvement and metrics. It also introduces you to the WISE tool. Chapter 3 talks about the Web Integrated Software Environment (WISE) tool in depth. The architecture

---

of the tool is laid out and many aspects of the design of each sub component is discussed. Chapter 4 gives a snapshot of WISE in action. This chapter gives one the opportunity to view the various facets of the WISE tool. Chapter 5 talks about other systems and readings related to my work. Chapter 6 concludes the thesis with some of my final remarks and possible future work. Appendix A provides the software internals giving one the essence of the coding that went into the making of WISE tool.

---

## Chapter 2

# System in a Nutshell

---

The World Wide Web Integrated Software Environment: WISE is a system for managing issue reports within a software development project. The tool enables managing 'to-do' lists among project developers and collects metrics related to changes in 'to-do' list items. Interlaced with metrics the tool provides efficient issue tracking<sup>2</sup>, performance indicators, coordination of work between developers and managers.

WISE is an automated tool based on WWW for managing access to project documents and of issue reports. Browsers<sup>3</sup> like Mosaic/Netscape and other user programs that can access the WWW can be used for this automating effort. Among the many advantages that such a automated effort would bring to a development group some of the key advantages are

- *Overcoming the geographical barrier:* By using a WWW client one can access another resource anywhere on the internet. This benefits the software teams who can access the tool from any place.

---

<sup>2</sup>Issue tracking can encompass more than merely problem tracking. One can deal with issues of all types.

<sup>3</sup>Browsers are versatile, multi-platform interface to the World Wide Web.

---



- *Overcoming the communication barrier:* Encourages collaborative software problem solving.
- *Problem Solving:* Software managers can effectively track the progress of their work group by using performance measures built into the tool itself. The different views the tool supports would give a better view of the database like closed issues, open issues and helps the managers handle the large amount of detail and help them in pinpointing early and suggest timely action.
- *Coordinating and maintaining data continuity:* Another advantage is that with such a tool is that changes in database are reflected on all views. There is a flow of data from one to another and updates are kept track of. This kind of continuous flow of data from the user to the manager and back, helps coordinate the works of many different people who work on common projects.
- *Collecting Metrics:* Collection of metrics is made an implicit process in the project management.

The critical resource being the data in the backend. A tool such as WISE provides means by which users can utilize the information in the backend database to help improve the software process from anywhere on the Internet using standard WWW browsers.

## 2.1 The need for software Process Improvement

---

The way we develop software has to be made more controlled and manageable process. In order to ultimately achieve the goals put forth, one must be in a position at any time to ascertain the quality of a software process. We must be able to improve the software process and determine if it is progressing at an acceptable rate. To produce products and improve the capability of the organization to produce better products, the software process must improve.

The software maturity framework<sup>4</sup> characterizes the different levels in which an organization can be categorized depending upon the results of the assessment of the organization's software process. Many case studies have been conducted and have shown that there is a need to turn the corner from chaotic, unpredictable cost to a more manageable and controlled software process whereby schedule slippage and cost overruns are avoided.

The effective use of software technology is limited by an ill defined process, poor process management. These problems have to be adequately addressed for us to be able to apply the software technology. For this the software process must improve. Many organizations have trouble defining and controlling their software process. An improvement in this can help them in transforming a user's need into an effective software solution [WATTS90].

---

<sup>4</sup>The process maturity framework is intended to be used with an assessment method. SEL derived this empirical model from the collective experiences of many software managers and practitioners.

---

## 2.2 Role of process improvement tools

---

A process description is a specification of how the job is to be done. It helps solve instances of the problems. Research is going on for effective process descriptions that are appropriate for the specification of the process which we use in developing and evolving the software. Process programming is the activity of expressing software process descriptions with the aid of programming techniques. This technique provides a vehicle for the materialization of the processes by which we develop and evolve software [LEON87]. In materializing software process descriptions we are creating a natural subject of measurement. Testing, Development can be thought of as execution of a process program. We can measure the degree of completion of the process in terms of the position of the execution pointer in the process program code. Knowing the overall process program and the current position our filter one can compare the status of project with previous metrics and indicate if the current software process is doing better than the others or not. Knowing this we can say if our software process has improved or not. WISE helps automate collection of metrics by making it an implicit process in project management.

Many organizations are now forming a software engineering process group that serves as the focal point for software process improvement. The main purpose of such groups is to establish and maintain the software process databases, initiate the definition, collection, analysis of process data. WISE automates the process of gathering and entering the data into the database, controls access to the database, supports

---

project in analyzing the data. The automation is prevalent at all stages of the project and helps by not only providing metrics not only the current state of the software process but also managing the issues and to-do items of the development team. Automated tools like the one mentioned above play an important role from the project inception to project completion.

## 2.3 Shaping the WISE project Management Tool

---

In shaping an automated tool to help in software process improvement and helps in change management we used World Wide Web as the framework for information exchange. The World Wide Web is based on client server architecture. The WWW consists of documents, links, indexes which may be searched. The WWW is becoming one of the largest byte and packet movers. The World Wide Web brings us the advantage of client/server architecture<sup>5</sup> without the infamous high cost of implementation and support. Another advantage of using the WWW is that application is being stored at its source, one can make frequent updates. The Web has brought a whole new meaning to the concept of *rapid application development*. The Web has become the application platform of choice for many enterprises [DOUG94].

WISE began as an idea to put a programmer's "to-do" list on the WWW and allow programmers to view the metrics. While making design plans for such a tool, we came across many discussions on the html work group for the need of an issue-tracking tool for HTML 2.0 specifications. A need for a tool would embody a mechanism to maintain a

---

<sup>5</sup> Web Client sends requests for document to any Web Server. Web Server upon receipt of a request sends the document requested back to the requesting client.

---

tree of all issues was needed for large projects. The idea of having the status of all issues in a nutshell lead to metrics. These metrics aimed at providing guidance for better decision making. The *availability* of WWW browsers and *accessibility* across all platforms make such a tool highly versatile.

## 2.4 Prototypes

---

Several programs and libraries exist that provide form interface in Mosaic to SQL databases. GSQL-ORACLE is a slightly modified version of the same software allows Oracle backends. The working of the tool is as follows: GSQL parses the forms based on some specification files we write and creates SQL statements. Then GSQL invokes the oracle backend with the SQL query and the arguments filled in by the user. The backend processes the query and returns the HTML documents. These results are sent to the client running one of the standard browsers. The limitation of this package was that it provided only select queries. It couldn't be easily molded to the application I had in mind. The high level language was weak to handle software process descriptions.

Another tool that was a candidate for WISE was WebStar that exploits the WWW environment and CORBA<sup>6</sup> environment. Web\* software allows the linking of any information source to a Web client such as Mosaic by allowing a person to specify HTML template that is dynamically filled in when requested by the user. The templates embed

---

<sup>6</sup>Referred also as Object Request Broker, the ORB is the communication heart of the standard. It provides an infrastructure allowing objects to converse, independent of the specific platforms and techniques used to implement the objects.

---

TCL commands and are interpreted and can be used to retrieve and dynamically fill the templates with information. Web\* provides mechanisms to deal with the stateless nature of the HTTP protocol. We implemented a prototype tool using this software but were not very comfortable with the dependencies the tool attached.

It is at time that we decided to build our own tool that would use the WWW as the framework for information exchange. It meant writing an Oracle gateway<sup>7</sup> to address the database connectivity. With such an approach WISE Alpha was born.

## 2.5 Integration of Metrics

---

Software Metrics can be defined as *continuous application of measurement based techniques to the software development process and its products to supply meaningful and timely management information together with the use of those techniques to improve that process and its products.* [LINDA94]. Software metrics are standardized ways of measuring the attribute of software processes, products and services.

The Goal, Question and Metrics Paradigm - GQM [VICTOR92] approach is used to select metrics. Instead of collecting metrics at random that usually becomes incomplete, inconsistent and inadequate, metrics collection based on the users of the tool embodies facts such as who needs the information, who is going to use the metrics, and the same could be used for functional and project management. The metrics

---

<sup>7</sup>Gateways are programs that handle information requests and return a document or generate a document on the fly. The Oracle gateway serves information to the client from a backend Oracle database.

---

featured in the WISE tool have a specific and defined objective. Measuring software is a powerful way to track progress towards project objective.

Metrics can be used to **understand** a software process, **evaluate** a software product and goals, **control** resources and products, **predict** attributes of the software process in the future. A database stores all the issues and the state of the issue. Queries that can extract important information relevant to the metrics feed into the metrics engine that provide the graphical front end.

## 2.6 Overview of System Working

---

The user of the tool specifies the URL (Universe Resource Locator) of the information server, logs into WISE, views their "to-do" list. Each person in an organization has a TO-DO list under WISE. Each item on a person's TO-DO list is called an issue. WISE supports 'issue tracking'. It encompasses problem tracking. WISE is intended to deal with issues of all types. The tool could be configured to be used by a system administrator to maintain a list of tasks to be done (to do list), by managers to track the status of various parts of a project, software engineers to maintain problem reports. An issue is any *action item* within an organization assigned to specific individuals. Each issue is composed of fields that include:

- The issue number
  - The project name of the issue
  - The issue description
  - The issue priority
  - The issue's status (new, open, fixed, closed)
-

- The issue's author's name

Issues in a TO-DO list can be sorted by their priority or by their status. Issues appear on a TO-DO list based on an individual's role within the organization. From the to-do list, you may also view project metrics. The WISE system keeps track of changes to issues and other project events. WISE collects metric based on these events and presents graphical views of the project statistics. For example, issues cycle from the open state, to the fixed state, finally become closed issues. Project Managers can view the current number of *open Vs closed* issues at any time in a project's life cycle.

Many different browsers can access WISE through the World Wide Web. WISE is best viewed using Netscape Navigator Version 1.1 of Netscape Communications Corporated. WISE provides metrics on projects in the form of graphs that compare aspects of the group activities. The system can answer queries about the evolution of issue reports in the system. WISE can be programmed to control the visibility of their metrics to managers and others. WISE is non intrusive because it provides a "to-do" list of each issues to each developer in the team. Each issue in the "to-do" list can be acted upon which change the issue status. The composition of the forms and views defines the totality of the software process. Thus, the process is not fixed or globally defined by the manager, but it is dynamic and changes based on the roles of development personnel.



## 2.7 Project Environment

---

To run the WISE project management and metrics tool following platform and tools would be needed:

- Solaris 2.3 or greater
- Oracle version 7.0
- Access to Oracle runtime library.

The tool is best viewed using Netscape version 1.1. Any browser can access the WISE tool. The httpd<sup>8</sup> server runs on the same machine that provides the backend database. The WISE tool makes use of GNUPLOT (© Thomas Williams, Colin Kelley) an interactive command driven function plotting program. PBMPLUS package (© Jef Poskanzer) provides programs to covert pbm formats to gif. GIF stands for Graphic Interface Format, property of CompuServe. It is a PC standard due to its compression method, Lempel-Ziv algorithm, a very efficient lossless one. PBM is the easiest and most common format among UNIX systems.

---

<sup>8</sup>The language that Web Clients and servers use to communicate with each other is called Hypertext Transfer Protocol (HTTP). Web servers are called http servers.. All Web Clients and Servers must be able to speak HTTP.

---

---

## Chapter 3

# WISE Architecture

---

WISE is based on a client server architecture. The World Wide Web is used as a framework for information exchange. WWW brings us the advantage of client/server architecture at a very low cost. Any web based application has some key components. One of them is Hypertext Markup Language or HTML. HTML is the SGML DTD (Document Type Definition of Standard Graphics Markup Language). HTML is a collection of styles used to define the various components of the WWW document. The chief power of HTML comes from its ability to link regions of text (and also images) to another document (an image). these regions are highlighted by the browser to indicate that they are hyper links. In WISE hypertext links are used to point to each issue. The Web also specifies a Hyper Text Transfer Protocol (HTTP) which supports a large number of asynchronous requests from multiple sites, the Web Server, also referred to as http server. Scripts written in CGI (Common Gateway Interface) allow the server side of client/server WWW applications to communicate with existing systems.

---

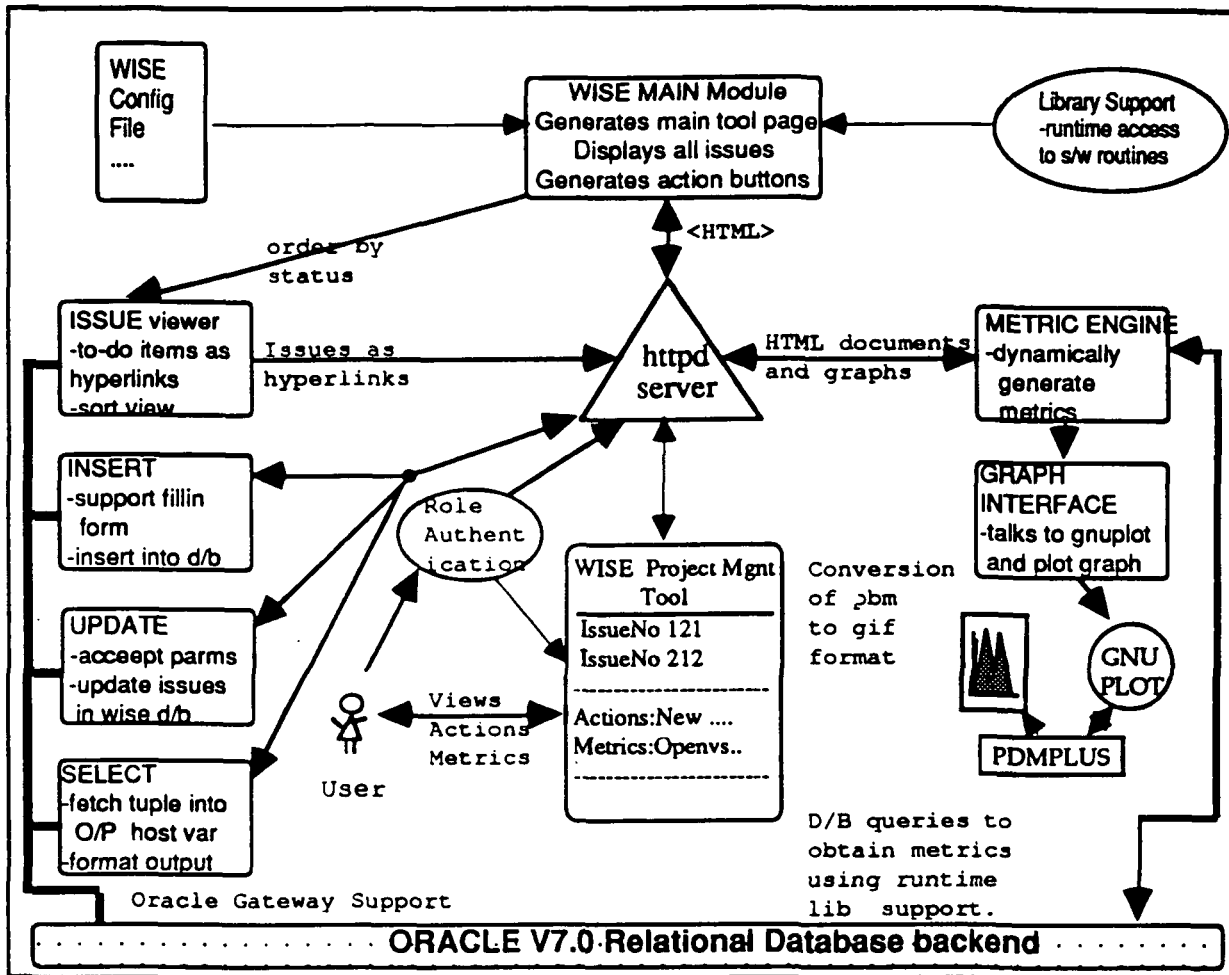


Fig 1. WISE Architecture

### 3.1 Overview of Tool Working

A URL (Uniform Resource Locator) refers to the format used by WWW documents to locate other files. A file on the WWW server is located using a scheme *http*. The user of WISE specifies the URL of the information server and views the "to-do:" list. Access is restricted using the HTTP/1.0 based authentication.

The user may perform one of the many operations like logging a new issue or acting on existing issues. They may also view project metrics.

## 3.2 Supporting To-do Items

---

WISE provides a means to view the to-do list items as hyper links. These to-do items are generated dynamically. The wise main module generates the main tool page, displays all the issues and generates all the action buttons like *new issue*, *help*, *metrics* like *open Vs closed*, *open Vs fixed* and more. The page also includes a Refresh HTTP response header. The individual refresh directive is one shot and non repeating. By suitably defining the refresh directive one achieves continuous loading, thereby making the tool's front end dynamic and presents a snap shot of the updated database.

In order to support the functionality of the main module we have a ISSUE viewer module that queries the database for all issues. For each tuple that is generated as a result of the query, the host variables in the program are properly initialized to hold each of the returned fields. Using hypertext related directive `<a href>`, each issue is made to be a hyper link. The text that will serve as the hyper link in the current document is properly formatted. The document that is being pointed to by the hyper link is another executable with parameter as its `issue_no`. This way each hyper link refers to the same backend executable but with different parameters (issue no is a primary key in the backend issue database). All action buttons are generated as hyper links.

### 3.3 Database Connectivity

---

The Common Gateway Interface (CGI) is a standard for external gateway programs interface with information servers such as HTTP servers. Gateways are really programs which handle information requests and return the appropriate document or generate a document on the fly. With CGI a server can serve information which is not in a form readable by the client (such as a SQL database), and act as a gateway between the two to produce something which the clients can use.

The Oracle gateway conforming to the CGI specification has been written using C and embedded SQL statements with Oracle run time library support and an Oracle PreCompiler. The gateway programs help by being able to store the issues in a relation database backend, and the power to perform *select*, *update*, *delete* queries too. The WISE gateway provides a form interface in a WWW browser to SQL database (Oracle v7.0) . Components of the fill-in form are mapped to host variables that are used in conjunction with embedded SQL statements. Cursor based control is used to fetch tuples from the backend. Since the makeup of the SQL statements are known before runtime static SQL statements serves our purpose. The gateway programs are placed in someplace accessible by the httpd server (usually the cgi-bin directory). The username and password are hidden from the client and thereby making information necessary to access the database restricted.

The Oracle PreCompiler is a programming tool that allows embedded SQL statements in a high level source program. The PreCompiler accepts the source program as input, translates the embedded SQL statements into standard Oracle runtime library calls, and generates a modified source program that one can compile, link and execute.

---

This lets us use the power and flexibility of SQL and combine its declarative style with the procedural style of programming languages like C. Various data manipulation SQL statements like SELECT, UPDATE, DELETE combined with transaction control statements like COMMIT, ROLLBACK give the host language the power to transfer data between Oracle and a host program. The host program connects to Oracle using executable SQL statements. All oracle objects, communication areas and SQL variables are handled using declarative SQL statements. The WISE tool comes with gateway support to query the backend database, update and delete issues in the backend database. Specific roles are attached to the users of tool and access to these programs are limited depending upon the access permissions.

### 3.4 Statistical Analysis Support

---

WISE generates lots of statistics on the existing process state using the backend database, which is the repository for all the logged issues. In order to support retrieval of metrics from this backend a metric engine was written. The main function of this engine was to generate statistics for a desired metric. The module includes code that makes calls to oracle run time library and using a cursor based approach obtains the relevant information. For e.g. open Vs closed is a metric which requires answers to two questions: how many open issues are currently present? how many closed issues are present? The module generates a macro file with all the data that completely answers the metric in a form that can be suitably used with gnuplot.

---

### 3.5 Graphical Support

---

Gnuplot is a command driven interactive function plotting program. The software was originally developed by Colin Kelley and Thomas Williams. For providing a graphical front end to all the metrics dynamically generated we needed a program to produce publication quality plots. For this purpose we initially thought of using `gd.h` - a graphical library available on the public domain. This library found use in generating statistics for Web Servers. In order to use the same library for generating graphs depicting the process state, we would have to write a complete class of graphical functions to generate graphs for various types. To overcome such an effort, we decided to use gnuplot. Gnuplot provides the flexibility of generating data files on a variety of terminals.

To overcome the interactive nature of gnuplot, input is provided through a macro file generated just after the database is queried. By specifying a terminal of type `pbm`, we used gnuplot to generate a `pbm` file. A public domain software called **PBMPLUS** © provides programs to convert the above `pbm` format to `gif` format. The resultant file is displayed by the Web browser as an inline image.

---

## Chapter 4

# WISE in action

---

### 4.1 Life cycle of WISE

---

Early in July '94 we worked on a software action plan for a small research group headed by Dr. Callahan. The purpose of the action plan was to be able to improve the software process and define how issues and change reports would be handled in an organization. Very soon after my completion of this draft, we started getting involved in process tools which would help in such an effort. We studied case tools and process improvement paradigms. We also started reading up lots of materials on software process and methods to improve them. Later that semester I presented a term paper on "*Issues relevant to Support Tools for V&V for the improvement of the Software Process*" as a part of the Advanced Reuse Course. It was at this time that the World Wide Web was gaining momentum and we developed a prototype tool using CGI. In December '94 the Software Engineering Process Group met and we decided to have the list of all issues and change reports as a "to-do" list. This concept changed the facet of the tool. It was a very important concept that revolutionized the purpose of the tool. The to-do list could

---



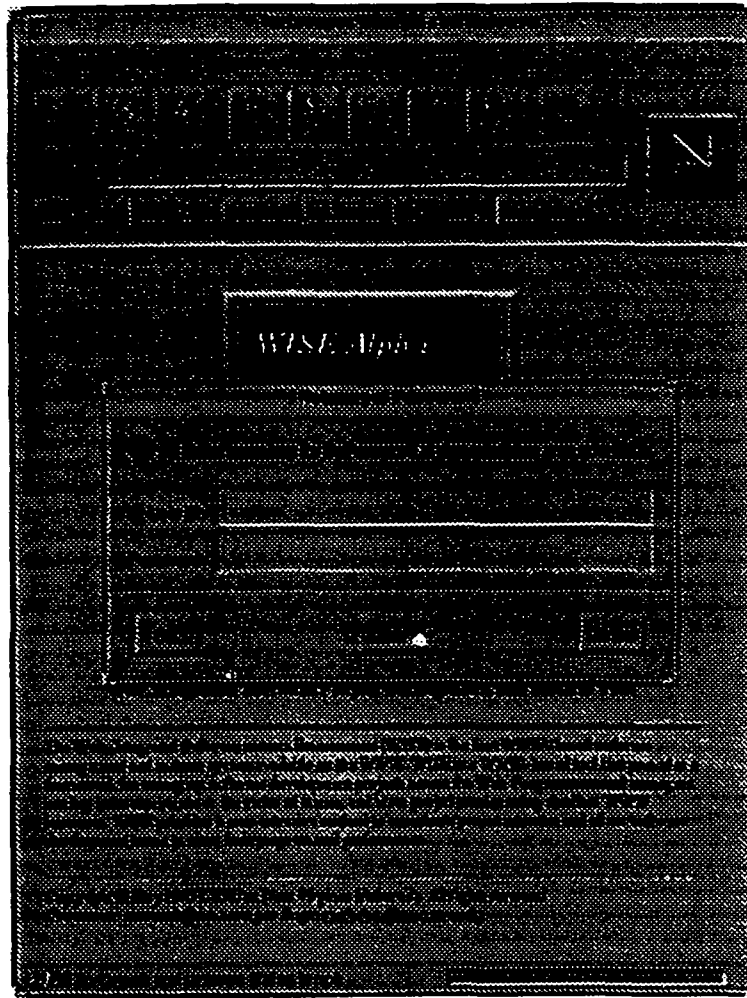


Fig 2: WISE Startup

take the place of not only change reports, bug-reports but issues. It was around that time that I continued my work on Metrics and was reading Dr.Basili's papers on Goal, Question, Metrics. Later in January'95 we searched the Web for public domain tools and found a tool called GSQL. We implemented a prototype version for issue viewing using GSQL in February'95. Not satisfied with the results , we used WebStar another public domain software. The dependencies the tool attached and not satisfied with the interface we decided to develop my own tool in March'95. Parallel to the work on the tool, we started working on the WISE Home Page. Late in April'95, we completed the

conceptualization and demo of the tool. In middle of May'95, we made the release of WISE public. We had a home page with extensive documentation, tutorials and proposals for Alpha sites. The responses and the need shown for such a tool was overwhelming. The WISE Home page has on an average 4000-5000 hits every week. Meanwhile in June we completed the pre-alpha release of WISE. WISE is currently being used to build WISE. It is also going to be used by the RMP group and a number of companies which serve as an Alpha Test site for the tool. WISE is also the first project management and metrics tool based on the WWW.

## 4.2 Startup

---

Assuming that the user has specified the URL of the information server, the user logs into wise by specifying a name and password. The user now selects a role. Roles that are supported in WISE fall into the three categories:

- *Developer:* A developers view of to-list shows him all open issues. The developer is allowed to log new issues, view project metrics. However he may update an issue to fixed only. He is not authorized to close a problem.
- *Administrator:* An administrator can view all fixed issues and is authorized to close an issue if he/she feels that the issue has been addressed and a solution to the same has been found.

The screenshot displays the WISE Main Page interface. At the top, there is a header area with a logo and navigation links. Below this, a table lists project issues with columns for Issue ID, Description, Status, and Date. The table contains several rows of data, including issues related to 'Building new software', 'New feature requests', and 'Software bugs'. Below the table, there is a section titled 'ACTIONS: How to use (Help)' which lists various actions such as 'NEW ISSUE', 'CLOSE ISSUE', 'EDIT ISSUE', 'DELETE ISSUE', and 'SORTING'. At the bottom, there is a footer area with additional information.

Issue ID	Description	Status	Date
101	Building new software	Open	01/17/91
102	New feature requests	Open	01/17/91
103	Software bugs	Open	01/17/91
104	Building new software	Open	01/17/91
105	New feature requests	Open	01/17/91
106	Software bugs	Open	01/17/91

**ACTIONS: How to use (Help)**  
 NEW ISSUE: Click on 'New Issue' (Open or Fixed Issue)  
 CLOSE ISSUE: Click on 'Close Issue' (Open or Fixed Issue)  
 EDIT ISSUE: Click on 'Edit Issue' (Open or Fixed Issue)  
 DELETE ISSUE: Click on 'Delete Issue' (Open or Fixed Issue)  
 SORTING: By Status (by Date) By Priority By Status

This is an example of a WISE TO-DO list. WISE allows programmers to view their work as a TO-DO list of issues. TO-DO lists are prepared for each individual in the organization.

This TO-DO list shows the issues that they should work on for their job as Project Manager. It is used to track and monitor their completion. If the user is not a Project Manager, they will not see this list.

Fig 3. WISE Main Page

- **Other:** In this role all the user can do is see all the open, fixed, closed issues. He/She is not authorized to update any issue. The user is provided with project metrics.

The user-id's are attached to the various roles in a software development group. WISE performs role-authentication for security purposes before the user is allowed access to the to-do list page corresponding to his/her role.

**Issue Report 255**

Project	ACTSAC
Description	On-line support tool
Module(s)	System
Priority	High
Status	Open
Owner	System
Author	System
Address	The system is the system support tool for the system support tool.
Filed by	System
Category	System
Severity Level	1
Test Case	255, 256, 257, 258
Issue Number	255
Parent Issue	255

**ACTSAC: Issue Report 255: Issue 255: Issue 255**

This is an example of a WISE issue report. Every issue has a unique ID. The following are the fields in this issue:

- Issue ID: 255 (This is the issue ID. It is the issue ID.)

Fig 4. WISE Issue View

### 4.3 WISE Main Page

After successfully logging into the WISE system, the user views his to-do list using a WWW browser. The display is dominated by issues seen as hyper links.

A number of action buttons are provided so that the user may act on the issues he views. Issues can be sorted by their status or by their priority. From the to-do list the user may view project metrics too. Each hyper link when clicked brings up the details pertaining to that issue.

## 4.4 Viewing an Issue

---

Each highlighted link on the main tool page behaves as a hyper link and clicking on it causes the details of that particular issue to be seen on your browser. The issue report generated has many buttons (hyper links) using which one could modify the state of an issue, update the action fields, or even delete an issue. The kind of privileges that a user is entitled to do depends on his/her role. Editable fields are marked. Using the navigational links at the bottom of the page one could leave the issue alone and return to the to-do list.

## 4.5 New Issue

---

Selecting the new issue hyper link in the main tool page causes the new issue form to be generated. The form displays an empty issue form. Filled with many radio buttons and menus the user logs a new issue by specifying the project, a brief description of the problem, expected due date. By default all new issues are assigned an open state. An action section appears in the issue only when the issue has changed state to fixed or closed. Every issue is identified by a unique number. Sub Issues of

---

NEW ISSUE REPORT X3	
Project:	<input type="text"/>
Issue ID:	<input type="text"/>
Module:	<input type="text"/>
Priority:	<input type="text"/>
Status:	<input type="text"/>
Date Enter:	<input type="text"/>
Action:	<input type="text"/>
Abstract:	<input type="text"/>
Find By:	<input type="text"/>
Category:	<input type="text"/>
Severity:	<input type="text"/>
Time Cost:	<input type="text"/>

Fig 5. WISE Issue Insertion

existing issues can be created by selecting the hyper link *create sub issue* when viewing an issue.

The header indicates which issue number the issue has been assigned. For new forms most of the fields are left blank. Scrollable menus, radio buttons and text entry fields are provided for ease of use. The user can edit some of the fields but not all of them. Some of the fields take on configurable values. After filling in the various sections of the form, hitting the submit button causes the information to be stored away

in the WISE database. WISE will validate the fields when you submit it and reject forms with invalid field values.

## 4.6 Viewing Project Metrics

---

Users may select one of the many metric hyper links from the WISE main page and view various metrics which are implicitly collected during the software process. Selecting the *progress* hyper link for instance causes the following metric page to be displayed. All data are dynamically generated. The user can compare the relative progress made on a per week basis and view various attributes of the software project. WISE also stores the actions taken on each issue in a backend database. This implicit storing of all actions taken on each issue helps in later analysis of the life cycle of issues.

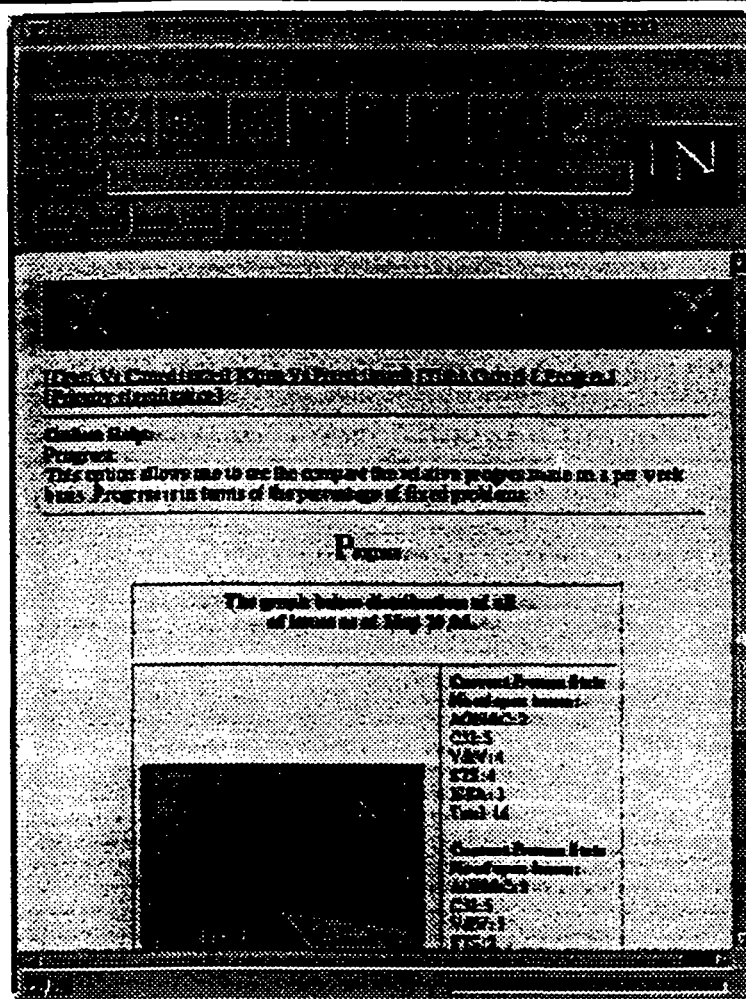


Fig 6. WISE Metrics



---

## Chapter 5

# Related Work

---

Do you believe in software automation as the key to increasing productivity, controlling quality, and introducing predictability into the software process? Below we summarize a lot of the work done by many in this area which convinced me into believing that automation in software management and measurement is a must if the software process has to improve.

### 5.1 Software Automation and Measurement-Strong as Ever

---

As a part of the Naval Research Laboratory (NRL) NASA's Software Engineering studies, results on collecting valid software engineering data is presented [VICTOR84]. The data to be collected were based on the changes made to the software during development. It was later followed by evaluating the software development by analysis of change data [WEB85]. WISE is an approach to improve the software process, collects useful data as changes are made to each issue and stores them in a backend database and dynamically upon request from the client generates metrics. Clients can view the

---

software engineering issue data. This data basically reflects the life cycle of all issues and various metrics can be derived to project useful project attributes.

Experience factory mechanism [VICTOR93] is an approach to improve the software development. The Experience Factory packages the experiences and measures of various software processes, products. The focus of such an approach is to support project development by analyzing and synthesizing all kinds of experience, acting as a repository for such experience, and supplying that experience to various projects on demand. WISE focuses on such an approach by collecting valuable information related to issue solving in a database implicitly. This information throws light on the issue solving capacity of the development group.

Software Metrics are standardized ways of measuring the attribute of software process, products and services. Software Measurement Services provide clients with many tutorials to design metrics. Tutorials provided by them [LINDA94] help clients establish data collection mechanism, design metrics, and most importantly help them to understand software metrics concepts. An excellent mechanism to defining an objective based measurement program is the Goal-Question-Metric (GQM) paradigm defined by Basili and Rombach [VICTOR92].

A metrics program [SHAR93] should address corporate needs and it should include linking metrics, to process maturity, a tools-evaluation database, and the use of multiple-metrics graphs. It is highly recommended to embed metrics tools in the existing development environment. Developers *should understand the need for metrics*, else they will neither provide accurate data or use the results of metric analysis. Secondly, metrics should be kept close to the developers. This way the developers would be able to access measurements, evaluate them, and take action as part of standard

---

operating procedure and without hindering schedule or budget. WISE address this aspect very well. The metrics are generated implicitly and can be viewed at any stage of the project by all developers. A separate metrics group is not present that collects metrics and analyzes them. The time devoted to metrics collection and analysis is minimized by WISE. The front end of WISE is kept very simple so that developers need not become experts in measurement theory. WISE generates graphs to amplify certain aspects of the stored data. The users are not burdened with the task of metric collection. WISE does it for them.

It is important to remember that metrics can only show problems and give ideas as to what can be done. It is the actions taken as a result of analyzing the data that bring the results. This is the reason why it is critical for metrics users to understand that measurement is not the goal. The goal is improvement, through measurement, analysis, and feedback [MIC92]. In the same paper a practical view of software measurement that formed the basis for a company-wide software metrics initiative within Motorola has been described.

Carma McClure, in her article "*The CASE Experience*" discusses some of the various CASE tools, tool kits and work benches that were available six years back [CARMA89]. The point that was being stressed is that through three experiences she shows how tools automate the software process. Her message at that time was "*by all means start now*". Software automation is nowadays playing an important role in defining software quality.

The software process program must be defined in a precise powerful and rigorous formalism. And such an environment would become a vehicle for the

---

organization of tools for facilitating development and maintenance of the specified process [LEON87].

There are many advantages of behavioral descriptions to describe software process [LLOYD88]. He talks about the spiral model. The behavioral approach describes software development as a collection of activities or processes which may take place concurrently. This approach leads to better automation, message passing for communication, provides greater visibility for software process. One should try to be able to describe the software process in terms of the events which occur in the development effort rather than the date of the product. WISE addresses this by using the events which occur, to track the issues.

Intelligent Assistance for Software Development and Maintenance is a something we vision WISE to be in future. WISE would need some intelligent engine to check the semantics of each issue and see if a similar issue exists in the database. Marvel is a tool that understands users action and sequences [KAIS88]. Marvel helps in programming by performing early error checking. Marvel helps user by informing him/her which components are potentially affected before actual editing. Marvel maintains knowledge about the specific development effort. In short tools such as marvel assist development and maintenance efforts through controlled automation.

Software tools play an important role in the software development process. Some tools are used by users taking on a specific role. Other tools are used by users in multiple roles, during many activities and for processing documents of multiple users. Inserting an tool could have a significant impact on the development process. In order to control the insertion of the tool a method called "Tool Insertion Method" [TIM94] has been proposed. The key elements of TIM are tracking the progress of a tool used to

---

improve the process. Coupled with such a tool WISE's impact on a software process can be analyzed.

Internet particularly the WWW should be the application platform of choice for many enterprises [DOUG95]. The internet is more cost effective than a private WAN. The architecture of WWW applications is conducive to rapid application development and provides network transparency.

WISE supports issue tracking. The issue based approach is not dominant only to problem tacking, but is critical in capturing design rationale, informal information. Design dependencies can be represented in a issue based style [MITCH81]. Participants in an issue -based discussion contribute their expertise and viewpoints to discover and resolve issues. Each issue is followed by one or more positions that respond to an issue. The issue based model is now almost 20 years old.

All these articles and papers show the spectrum of work being done and opportunities for further improvement become clear. A need for an automated tool for software management and measurement is created by the fact that focus on understanding and managing the software process is the place where one can expect significant improvement.

## 5.2 Systems

---

WISE is the first project management and metrics system based on the World Wide Web. We did happen to learn about some of the other tools related to issue tracking, defect managers, and problem trackers. None of the tools had an implicit metric system built into it. We would like to summarize a few of the many tools

---

available on the Internet. Some of them are commercial versions and have some nice features which if included in WISE would make it even better. In the paragraphs to follow we shall describe in brief some related work to the tool we built. Some of the systems overview described have been taken from the config management FAQ [DAVE95].

### Problem 1.1:

Problem is a problem tracking tool, a "problem" database manager. It is a database manager for bug reports and such, meant to be used in the UNIX environment. It is written in C++; uses the GNU Database Management Library (GDBM) for low level database operations; and the termcap library for screen control. An X Windows interface is being developed. The basic idea is to provide a central front end for managing various databases of bugs that a large UNIX site might be interested in tracking, and facilitating the sharing of information amongst all interested parties. We used the tool couple of months back. The tool though not intended to support metrics lacks support for measurement. Secondly the view of bug reports is not user friendly. The user has to remember very minor facts like issue number. The level of automation is restricted. It however does provide e-mail connection amongst all group members which my tool lacks. Also the concept of having many databases is currently not supported in my tool but is not difficult to add. The problem 1.1 can be improvised to a great extent. But its accessibility would be restricted to users in the UNIX environment. The user interface and GUI support for the current version is not comparable to WISE. In our opinion WISE out performs problem 1.1 in all respects, both tracking issues and providing metrics for software process improvement.

---

## Gnu Gnats:

Gnu Gnats (GNU Problem Report Management System) is a product that tracks software problems or change-requests. Some of its features include:

- problem submitted via e-mail
- uses a file system database
- each problem identified by a unique key
- querying possible
- can maintain audit trail of all activities concerning a specific problem

GNU GNATS does not support implicit metric collection. GNU GNATS supports problem tracking and not issue tracking. WISE in our opinion supports much better functionality's as compared to Gnu Gnats.

## DDTs:

DDTs by Qual Trak is supposed to provide efficient problem management and metrics gathering. E-Mail interface allows notification of new problems and changes of status. DDTs provides a simple query language that allows extracting of various project statistics. Users could create their own scripts for specialized queries. The FAQ posting of problem management tool summary describes this tool as having received the second highest number of recommendations. It is described as a very flexible tool. It supports parent-child relationships. The biggest disadvantage is that it is available on the UNIX platform. WISE in comparison doesn't as of yet provide the user with the flexibility to create customized queries like what DDT supports. WISE also doesn't have an e-mail support. But WISE is a public domain software and uses WWW as the framework for information exchange. The accessibility of the tool across all platforms

---

makes WISE more versatile. WISE also supports the parent-child relationships and would in future be integrated to generate the issue-hierarchy tree with the DOT software.

#### Scopus:

The tool includes problem management (QualityTEAM). Provides bug tracking and easy to add/modify the system screens and functions. The tool has a GUI interface and works on the Sybase database. The customization is supposed to be very easy. Scopus is believed to one of the more popular problem management systems. WISE in comparison is accessible across all platforms, supports parent-child issues. WISE doesn't have features like ad-hoc query builder or keyword search that Scopus has. Keyword search is one of the features we have cited as my future work.

#### Razor:

Razor from tower Concepts, Inc. supports problem tracking. The products supports many features that WISE supports. The heart of the Razor package is issue tracking. The product supports e-mail connectivity which WISE lacks. WISE may out perform their tool by cost and being based on WWW provides access across any platform to the issue database.

#### PR Tracker:

PR tracker is a shareware program developed by Softwise that helps manage software development projects by tracking software bugs with problem reports. They currently have a Windows beta version. PR records problem reports in a network database, supports simultaneous access to database by multiple users, helps estimate the time

---



required to complete the projects. WISE lacks such estimation engines. But supports powerful metrics.

### QDM Defect Manager:

The QDM defect manager is a defect tracking and analysis system. It is a system which allows one to improve product quality by tracking, assigning, prioritizing, and analyzing product defects. It has graphically built SQL queries, added security, trigger notification, e-mail notification, handles dependencies among projects. The defect tracking tool is a new tool on Windows platform. WISE as before has a edge over the tool with respect to its accessibility across any platform.

### Others:

VTT Electronics has worked on metric programs on GQM bases for over a year and are about to specify the tool environment for collecting metrics and analyzing metrics data.

At the Artificial Intelligence Applications Institute, UK, work is going on to ascertain the value of issues/to-do lists or agendas as first class objects in activity models. They have put together all a collection of notes on <I-N-OVA> constraint model of activity, where I stands for issues. <I-N-OVA> is supposed to help by supporting work on automatic manipulation of plans, human communication about plans, principled and reliable acquisition of plan information, and formal reasoning about plans.

Capstone Corporation have developed a Software Change Request Tool based on Oraperl. We believe their product doesn't have the metrics facility WISE has.

---

## Chapter 6

# Conclusions

---

We began this thesis presenting a detailed study for the need of process improvement tools. WISE, an automated system helps in software project management and measurement. We have described how such a tool would help in software development life-cycle. An important aspect of software development "measurement" was given some focus. The WWW technology was introduced in the paragraphs that followed. Having introduced project management and metrics we then fused the two and gave a short overview of how a system which made metrics collection an implicit part of the software process. A architecture for the WISE tool was then proposed. It is based on the client-server architecture of the World Wide Web.

We went over some implementation related details. Many internals with regard to gateways and connectivity to a backend database were highlighted. The impact such a tool could have on project estimates and serving as a good performance indicator have also been highlighted. the various ways in which metrics data and analyzed and put forth in a way that the user can comprehend has a lot of influence on the overall quality of the software product.

---

Aspects related to various design and implementation issues were discussed in the paragraphs that followed. Related work to WISE was covered in great depth. A brief overview of existing problem management tools was summarized. The life cycle of wise gives one a snapshot of the effort involved in the making of WISE.

In conclusion, we summarize, the major lessons learned:

- Automated tools that help in issue tracking can lead a project team to improve their software process during development.
- An important aspect of software development is measurement. Metrics help measure the attribute of software processes, products, and services. WISE helps in the analysis of process data.
- Such analysis focuses the team on the immediate experience of development and helps them correct process problems as well as validate whether those problems have indeed been corrected.
- A software process must improve. Use of automated tools such as WISE can complement current practices of in-process improvements.
- Statistical control in measurement of software process is of importance. All the metrics when analyzed and put in a proper way can help the V&V team give better insight into the current state of the project.
- While building support tools such as WISE, a framework like WWW helps in rapid application development, helps overcome geographical barrier in accessing such a tool across all platforms, and makes it highly versatile.
- Automated tools should make metrics collection an implicit part of the software process and should be able to provide metric analysis dynamically.

This helps the users of the tool ascertain their improvement and minimizes

---

the time to metrics collection. The developers should be able to access the measurements, evaluate them and take action.

World Wide Web serves as medium to make more and more information available on the internet. The WWW is now changing to a rapid application development platform. The critical resource being the data in the backend. It is necessary to utilize the information in a way that would help improve the software process. The concept of to-do list makes issue handling very easy and informal. Metrics collection is made implicit. WISE is an active issue solving system, allowing users to view project metrics, watch issues change state and most of all help them track issues. Through various action buttons and metrics the tool provides a comprehensive and effective means of handling issues and viewing project metrics.

The central issue in this thesis is the role of WISE as an automated support for software project management and measurement. It can be verified by the following:

- Users can use WISE to log new issues from anywhere in the Internet.
- Different users can interact on the same issue, thereby supporting collaborative software problem solving.
- Clients enjoy viewing project metrics. Measurement helps them to keep update on the project status.
- Users can view different metrics that reflect the current process state.
- Effective issue tracking.

WISE supports different roles in a development team. WISE currently supports three roles:

- Client role: The clients can use the URL of the server to log new issue. They may however be able to view project metrics and view all the issues.
- Development Role: This role is given to the developers and programmers in a developmental team. The users in this role have the added flexibility to update an issue to fixed. They are not allowed to close an issue or delete an issue.
- Superuser Role: this role is usually given to project managers who approve an issue and close it finally. They may also delete an issue.

Tools that help in software process improvement are becoming more and more popular. Even more are applications based on the World Wide Web. WISE has extended the state of art in information technology , project management, metrics and World Wide Web by exhibiting the following advantages:

- Web's ability to provide such benefits as shorter time to market, cost savings.
- Makes metric collection an implicit process in a project management
- Ability to view metrics that reflect current process state
- The system demonstrates an effective way manage software projects by making the interactions implicit and communications informal.

It is interesting to see the overwhelming response for such a tool. I am sure that this tool would pave the way for better tool that would enable software process improvement and we see a more controlled and manageable process.

---

This research talked about automated support for project management and measurement. The WISE system is an approach to solve the problems current software development groups face and provides an cost effective way to overcome the hurdles.

We have completed the Alpha version of WISE. The WISE system has its limitation. One of the biggest limitation is that the current version works only the Solaris platform. the tool has to be ported onto many other platforms. We haven't yet tested the system on other platforms.

Several steps should be taken to make the system easily configurable to the users needs. Currently the configuration is restricted so that the user could download the source code and have an identical version of the tool running. Explicit entry into the source code has to be done to adapt the tool to a different environment where the software process differs. The solution is to have a high level language which would have constructs that describe some aspects of the software process like the nature of an issue report, the roles being supported, the kind of authentication support for each role. The ability to mold the layout of an issue should be configurable at a n high level. One solution currently thought of is the WISE programming language that would embed the software process itself. A WISE compiler would then generate the cgi code. The first step would be to automate the code level changes which have to be done manually at configuration time.

WISE's ability to define metrics is limited to a small subset of metrics. this means that some metrics which are not supported have to be written. The way the metrics engine has been written is such that it is easy for a third party to attach their metric program. A support for automating metric creation in my opinion would improve the efficiency of the tool.

---

Another drawback in the tool is that WISE doesn't have the capability to introduce new roles dynamically. A need may arise in a software process where an issue might have to pass through a new meta stage instead of the usual open to fixed to closed. The tool must be able to adapt itself to the changing process. If this adaptable feature were added to the tool, it would add the flexibility of trying out new process definitions.

Another problem is the semantic meaning of each issue. No support is provided to compare existing issues and avoid redundancy. A intelligent algorithm is required that would possibly compare some key words of each issue and trigger an alarm window to the client informing him that a similar issue already exists.

A related problem is the inability to define triggers from the database to the httpd server directly. We are currently using the client pull and server push mechanism provided by Netscape to dynamically update the document. This dynamic updating is necessary so that the user need not reload and any update to the database gets reflected on each of the browsers referring to the information server serving the WISE tool.

The famous problem of multiple updates at the same time and providing consistency has to be addressed in a better detail. We are currently using locks to prevent multiple updates of the same table at the same time. Other features that need to be provided is the ability to search the WISE database for issues that have particular keywords alone and in the programmers to-do list generate those issues that have the keywords. This would mean creating a separate configuration file for each user. This concept is very similar to creating virtual news groups with topics of interest to each

---

user [CURT91]. In the age of information modeling we think being able to model users interest is a very crucial point.

A possible extension to the tool would be using HotJava<sup>9</sup> Browser to make it object oriented. Each issue has an unique identity and being able to provide some properties to each issue that its child issue can share and thereby using the inheritance property to make the issue handling more powerful. The Java browser released by SUN is the next generation browser. Unfortunately the release date of Java was too late for us to make a change to Java.

Feedback agents need to be provided to all the Alpha testers of WISE so that future releases may support them. A project to improve the fit of process and tool and analyze the impact of tool insertion really caught our attention [TIM94]. The tool insertion method helps to improve tool/process fit and to analyze tool insertion impact. Before insertion of the tool, data is used to estimate the value of inserting a tool into the process. After inserting the tool the analyzed data can be used to assess the actual impact and value of inserting the tool. We wish the same could be done for the WISE tool in a software development effort and we could assess the actual impact of a software process improvement tool.

---

<sup>9</sup>HotJava is a World Wide Web browser that can execute programs written in the Java programming language and included in HTML pages. Applications written in Java can migrate transparently over the Internet accessible by anyone using the HotJava browser.

---



---

## Bibliography

---

- [ BARRY80 ] Barry M. "Airborne Systems Software Acquisition Engineering Guidebook for Software Testing and Evaluation", TRW Defense and Space Systems group, March 1980.
  - [ BUFE90 ] Martin E. Bush, Norman E. Fenton, "Software Measurement: A Conceptual Framework", Journal of Systems Software, December 1990, pp. 223-231.
  - [ CARMA89 ] Carma Mc Clure, "The CASE Experience", BYTE, April 1989, pp. 235-242.
  - [ CURT91 ] Curt Stevens, "Automating the Creation of Information Filters", Communications of the ACM, December 1992, Vol. 35, No. 12, pp. 48.
  - [ DAVE95 ] Dave Eaton, Config Management FAQ Editor, "Problem Management Tools Summary", <http://www.iac.honeywell.com/Pub/Tech/CM/PMTools.html>, part 3 of the FAQ.
  - [ DOUG 95 ] Doug Kaye, "The Internet as an Application Platform", Rational Data Systems White paper, May 1995. URL: <http://www.rds.com/whpapers.htm>.
  - [ IRENE92 ] Irene Stadnyk and Robert Kass, "Modeling Users' Interests in Information Filters", Communications of the ACM, December 1992, Vol. 35, No. 12, pp. 48.
  - [ KAIS88 ] Gail E.Kaiser, Peter H.Feiler, "Intelligent Assistance for Software Development and Maintenance", IEEE Software, May 1988, pp. 40-49.
  - [ LEON87 ] Leon Osterweil, "Software Processes are Software Too", ACM, 1987.
  - [ LINDA94 ] Linda Westfall, "Software Metrics That Meet Your Information Needs", Tutorial, Software Measurement Services, September 1994.
  - [ LLOYD88 ] Lloyd G. Williams, "Software Process Modeling", tenth International Conference on Software Engineering, April 1988, pp. 174-186.
  - [ MIL79 ] Miller E. "Automated Tools for Software Engineering ", IEEE Computer Society Press, 1979, pg. 169
-

- 
- [ MIC92 ] Michael K. Daskalantonakis, "*A Practical View of Software Measurement and Implementation Experience With in Motorola*", IEEE Transactions on Software Engineering, Vol. 18, No. 11, Nov. 1992, pp. 998-1010.
- [ MITCH81 ] Mitchell D. Lubras, "*Representing Design Dependencies in an Issue-Based Style*", IEEE Software, July 1991, pp. 81-89.
- [ NICH92 ] Nicholas J. Belkin and W. Bruce Croft, "*Information Filtering and Information Retrieval : Two sides of the same coin*", Communications of the ACM, December 1992, Vol. 35, No. 12, pp. 29-38.
- [ PRESS92 ] Roger S. Pressman, "*Software Engineering - A Practitioners approach*", Third Edition, McGraw-Hill Series.
- [ SHAR93 ] Shari Lawrence Pfleeger, "*Lessons Learned in Building a Corporate Metrics Program*", IEEE Software, May 1993, pp. 67-74.
- [ RAMA95 ] Ramakrishnan, S. and J. Callahan, "*WISE: The WWW-Integrated Software Environment*", NASA Technical Report NASA-IVV-95-001, NASA/WVU Software IV&V Facility, Fairmont, WV, June 1995.
- [ ROB81 ] Robertson, S. E, The Methodology of Information Retrieval Experiment, "*Information Retrieval Experiment*", in K. Sparc Jones, Ed. Chap. 1, pp. 9-31. Butterworths, 1981.
- [ SHOSH92 ] Shoshana Loeb, "*Architecting Personalized Delivery of Multimedia Information*", Communications of the ACM, December 1992, Vol. 35, No. 12, pp. 39-48.
- [ TIM94 ] Tilman Bruckhaus, "*TIM: A tool Insertion Method*", Proceedings of the 1994 CAS Conference, October 1994, IBM Canada Ltd. and the National Research Council of Canada, <http://spfteng.cs.mcgill.ca/~till/publications.html>
- [ VICTOR84 ] Victor R. Basili and David M. Weiss, "*A Methodology for Collecting Valid Software Engineering Data*", IEEE Transactions on Software Engineering, Vol. SE-10, No. 6, November 1984, pp. 728-738.
- [ WATTS90 ] Watts Humphrey, "*Managing the Software Process*", SEI series in Software Engineering.
- [ WEB85 ] D. M. Weiss, V. R. Basili, "*Evaluating Software Development by Analysis of Changes: Some Data from the Software Engineering Laboratory*", IEEE Transaction on Software Engineering, vol. SE-11, No. 2, February 1985, pp. 157-168.
- [ VICTOR92 ] Victor R. Basili, "*Software Modeling and Measurement: The Goal/Question/Metric Paradigm*", Computer Science Technical Report Series, University of Maryland, College Park, CS-TR-2956, September 1992,
-

- [ VICOTR93 ] Victor R. Basili, "*The Experience Factory and its Relationship to Other improvement Paradigms*", Lecture Notes in Computer Science 717, Software Engineering ESEC '93, 4th European Software Engineering Conference Garmish-Partenkirchen, Germany, September 1993 Proceedings.

---

## *Appendix A*

---

### Software Internals

The WISE program has many components. In this section we would be going over some of the main components that make wise and the internals of the program.

#### HTML

HTML stands for Hyper Text Markup Language. It defines the components of a World Web Document. Many of the programs in WISE generate HTML file that consists of text, and tags which tell the browser how to format the text. Documents can be enhanced using images, graphics, layout and appearance.

#### Form Creation

Fill out form support is implemented by having cgi-scripts generate FORM tags. FORM tags specifies a fill-out-form within HTML documents. With a combination of INPUT Tag, SELECT tag, TEXT AREA tag, and Form submission cgi-scripts generate new issue report forms which the user uses to complete and log an new issue.

---

## CGI libraries

In order to write CGI code using C, EIT has put out a CGI library to write better CGI programs. In order to pass data about the information request from the server to the script, the server uses command line arguments as well as environment variables. The environment variables are set when the server executes the gateway program.

## Gateways

Gateways are programs that handle information requests and return a document or generate a document on the fly. we have built a Oracle gateway that can serve information to the client from a backend Oracle database. The gateway is written in C program coupled with embedded SQL statements. By embedding SQL statements in the high level source program and subsequent pre-compilation SQL statement are converted to standard Oracle runtime library calls. By using indicator variables, host structures, and data types, cursors and transaction control statements one can write flexible applications. Many of the cgi scripts that talk to the database return the data into host output host variables and these can be processed using the procedural strength of C and the output can be formatted with HTML tags. This way the client sees the result of the queries properly formatted.

## Dynamic documents

The server push and client pull capabilities of Netscape add another dimension to WISE and makes the tool even more powerful. These features give WISE the additional strength of including directives in the CGI scripts output to reload the tool after a given number of seconds. The automatic reloading makes sure that the issues generated on the main tool page are a snapshot of the WISE backend database at all times. Also the

---

client pull mechanism is used to refresh the document with notices of further action. It is used specially for displaying prompt messages to the user like "Issue #213 has been deleted" and returns the user automatically to the main page.

### Authentication support

HTTP/1.0 based authentication is used to restrict WISE to user name/password level access authorization. Different roles are attached to different passwords.

### httpd server

The server that makes the hyper text and other documents available to the Web Browsers is release http/1.3 of NCSA httpd. The code is in public domain.

### Graphics

GNU PLOT is command driven interactive function plotting program. It is used by WISE to plot metric graphs and generate pbm code. The commands to gnuplot are contained in a macro file that are generated by the gateway program for the metric in question. The terminal type is specified in the macro file.

### Conversion from pbm to gif

PBMPLUS is a tool kit for converting various image formats to and from portable formats. The package includes some simple tools for manipulating portable formats. We use PBMPLUS package to convert the pbm to gif through a three step process:

```
pnmtowd | xwdtopnm | ppmtogif
```

The resultant gif file is projected on the clients page as an inlined image.

## Backend support

The backend is an Oracle relational database (version 7.0 Oracle). The table holds all the issues. The table structure varies depending upon the makeup of the issue.

## Generating issues as hyperlinks

In order to generate issues as hyperlinks instead of scroll bars, we have a module that talks to the database using cursor approach. For each tuple that is fetched, the module generated a hyperlink tag with a name of executable that would process the hyperlink along with a parameter that contains the primary key for the issue. This way every issue generated as an hyperlink would differ from the other only in terms of its parameters. A single executable takes the parameter upon clicking and queries the database to fetch the fields of that particular issue only.

## Supporting different views

In order to support different views filters that sort the way issues look have been provided. They basically process the fetched tuples and display them on a particular order.

## Help

On-line help is provided for each sub option and action buttons that a user views on the WISE page.

